TRATTATO LIFE OMEGA INFINITY — CASTINO EDITION

Struttura e limiti della stabilità nei sistemi aperti

Pierernesto Di Paolo
Ricercatore indipendente – Italia

Castino, 2026

DOI: https://doi.org/10.5281/zenodo.19314770

Diritti e utilizzo

© 2026 Pierernesto Di Paolo

Titolo: LIFE OMEGA INFINITY – Struttura e limiti della stabilità nei sistemi aperti

Autore: Pierernesto Di Paolo

DOI (versione principale):

https://doi.org/10.5281/zenodo.19314770

Versione specifica:

Versione v2 (29 marzo 2026)

https://doi.org/10.5281/zenodo.19315777

Archivio ufficiale:

Zenodo (CERN Open Science Repository)

Licenza:

Creative Commons Attribution 4.0 International (CC BY 4.0)

Questa opera è resa disponibile per la consultazione, la distribuzione e il riutilizzo a condizione di attribuzione dell'autore.

Il contributo presentato è di natura metodologica e teorica. Le eventuali applicazioni e validazioni empiriche sono demandate a verifiche indipendenti su dataset osservabili.

Prima pubblicazione:

2026

Tutti i diritti morali sono riservati all'autore.

Nota dell'autore

Questo lavoro nasce da una difficoltà concreta: descrivere condizioni di stabilità in sistemi che non sono in equilibrio e per i quali non è disponibile una descrizione completa della dinamica. Non introduce una teoria generale. Propone uno strumento operativo, ne esplicita i limiti e chiarisce in quali condizioni può acquisire contenuto empirico.

Testo

In molti sistemi reali la stabilità non coincide con assenza di variazione. Esistono configurazioni che persistono nel tempo pur attraversate da flussi continui. Il problema, quando non si dispone di una descrizione completa del sistema, è evitare di introdurre più struttura di quanta se ne possa giustificare.

Per affrontare questo problema si introducono due grandezze stimabili a partire da segnali osservabili: una quantità che rappresenta l'energia disponibile e una quantità che rappresenta la dissipazione. La relazione tra le due è espressa nella forma

C_T = E_Omega / DeltaS_dis

Questa espressione non è una legge. Il suo significato dipende dalle definizioni operative delle grandezze.

Nella tradizione della termodinamica dei sistemi lontani dall'equilibrio, la stabilità viene trattata attraverso la produzione entropica e i flussi. In quei modelli la struttura del sistema è il punto di partenza. Qui il percorso è inverso. Si parte da segnali osservabili e si costruisce una descrizione locale.

Questo implica che la stabilità non sia dedotta ma inferita su intervalli finiti. È quindi necessario distinguere tra stabilità orbitale e stabilità strutturale. L'indice introdotto appartiene alla prima categoria. Descrive la persistenza delle traiettorie osservate ma non fornisce informazioni sulla robustezza rispetto a variazioni della dinamica.

Quando si considerano sistemi accoppiati, la stabilità emerge dal mantenimento di relazioni interne nel tempo. Le frequenze osservate sono proprietà locali e non assumono valore universale.

Nel passaggio ai sistemi biologici le grandezze non sono direttamente accessibili. Si lavora con segnali derivati, tra cui gli intervalli RR. L'analisi è stata impostata su archivi pubblici di PhysioNet relativi a soggetti sani, a ritmo sinusale normale e a insufficienza cardiaca congestizia; nel primo archivio è stato utilizzato, come caso iniziale, il file identificato come "000.txt".

Le serie sono state suddivise in finestre temporali di durata costante. Su ciascuna finestra vengono calcolate grandezze derivate dal segnale.

La quantità E_Omega, nella forma implementata, è ottenuta tramite normalizzazione del segnale seguita da una stima di potenza media. Questa scelta produce valori relativamente stabili tra finestre.

La quantità DeltaS_dis è stimata come variazione dell'entropia spettrale tra finestre temporali adiacenti.

L'indice C_T è quindi calcolato come

C_T = E_Omega / DeltaS_dis

In questa formulazione, la variabilità dell'indice è dominata dalla variazione entropica. L'indice si comporta come una trasformazione inversa della dissipazione locale. Questo non introduce una nuova informazione rispetto alla dissipazione, ma ne modifica l'uso rendendo il confronto tra finestre direttamente interpretabile.

Segmenti con distribuzione più compatta degli intervalli RR corrispondono a valori più elevati dell'indice. Segmenti con maggiore dispersione producono valori ridotti.

L'estensione a dataset indipendenti mostra una coerenza qualitativa del comportamento dell'indice. Nel dataset patologico si osserva una maggiore dispersione e una riduzione dei valori dell'indice rispetto ai dataset fisiologici.

Il confronto tra dataset suggerisce una separazione tra condizioni diverse. La verifica quantitativa di tale separazione richiede l'esecuzione dello script su più soggetti e la valutazione statistica dei risultati ottenuti.

In alcuni segmenti si osserva una riduzione dell'indice associata a variazioni limitate di RMSSD. Per verificare questa osservazione è stata eseguita un'analisi di correlazione con ritardo tra la serie C_T(t) e la serie RMSSD(t + k). Su segmenti selezionati si osserva un massimo di correlazione per ritardi positivi.

Questo risultato è trattato come indicazione empirica e non come proprietà generale. La sua validità dipende dalla replicazione su più soggetti.

Per questo motivo l'analisi è estendibile a più registrazioni appartenenti ai dataset considerati. La validazione è considerata soddisfatta solo se la distribuzione dell'indice è coerente all'interno della stessa classe, se la separazione tra classi è statisticamente significativa e se una quota non marginale di soggetti mostra un ritardo positivo.

Finché queste condizioni non sono verificate su larga scala, il risultato resta locale.

Il contributo del lavoro consiste nell'introduzione di uno strumento replicabile il cui valore dipende dalla verifica empirica.

Bibliografia

- **Prigogine I**. (1967) Introduction to Thermodynamics of Irreversible Processes
- **Glansdorff P., Prigogine I.** (1971) Thermodynamic Theory of Structure, Stability and Fluctuations
- **Nicolis G., Prigogine I**. (1977) Self-Organization in Nonequilibrium Systems
- **Kuramoto Y**. (1984) Chemical Oscillations, Waves, and Turbulence Strogatz S. (2015) Nonlinear Dynamics and Chaos
- **Schneider E., Kay J.** (1994) Life and the Second Law

APPENDICE A — SCRIPT PYTHON ESEGUIBILE COMPLETO

```
import argparse
from pathlib import Path
import numpy as np
import pandas as pd


def rmssd(rr):
    rr = np.asarray(rr, dtype=float)
    if len(rr) < 2:
        return np.nan
    return np.sqrt(np.mean(np.diff(rr) ** 2))



def sdnn(rr):
    rr = np.asarray(rr, dtype=float)
    if len(rr) < 2:
        return np.nan
    return np.std(rr, ddof=1)



def energy_proxy(rr):
    rr = np.asarray(rr, dtype=float)
    if len(rr) < 4:
```

```
        return np.nan
    std = np.std(rr)
    if std == 0:
        return 0.0
    x = (rr - np.mean(rr)) / std
    return np.mean(x ** 2)


def spectral_entropy(rr):
    rr = np.asarray(rr, dtype=float)
    if len(rr) < 8:
        return np.nan
    x = rr - np.mean(rr)
    ps = np.abs(np.fft.rfft(x)) ** 2
    if len(ps) <= 1:
        return np.nan
    ps = ps[1:]
    total = np.sum(ps)
    if total == 0:
        return np.nan
    p = ps / total
    p = p[p > 0]
    return -np.sum(p * np.log(p))
```

```
def load_rr_file(path):
    rr = pd.read_csv(path, header=None).iloc[:, 0]
    rr = pd.to_numeric(rr, errors="coerce").dropna().to_numpy()
    rr = rr[rr > 0]
    t = np.cumsum(rr) / 1000.0
    return pd.DataFrame({"rr": rr, "t": t})



def make_windows(df, window_s=300, step_s=60):
    windows = []
    start = df["t"].min()
    end = df["t"].max()

    while start + window_s <= end:
        w = df[(df["t"] >= start) & (df["t"] < start + window_s)]
        if len(w) > 20:
            windows.append((start, start + window_s, w["rr"].to_numpy()))
        start += step_s

    return windows



def compute_metrics(windows):
    rows = []
    prev_entropy = None
```

```
for start_s, end_s, rr in windows:
    e = energy_proxy(rr)
    s = spectral_entropy(rr)

    if prev_entropy is None or np.isnan(s):
        delta_s = np.nan
    else:
        delta_s = abs(s - prev_entropy)

    prev_entropy = s

    if np.isnan(e) or np.isnan(delta_s) or delta_s <= 0:
        ct = np.nan
    else:
        ct = e / delta_s

    rows.append({
        "window_start_s": start_s,
        "window_end_s": end_s,
        "rmssd_ms": rmssd(rr),
        "sdnn_ms": sdnn(rr),
        "energy_proxy": e,
        "spectral_entropy": s,
        "delta_s_dis": delta_s,
```

```
        "c_t": ct
    })

    return rows



def lag_correlation(ct, rmssd, max_lag=5):
    results = []

    ct = np.asarray(ct, dtype=float)
    rmssd = np.asarray(rmssd, dtype=float)

    for k in range(-max_lag, max_lag + 1):
        if k < 0:
            x = ct[:k]
            y = rmssd[-k:]
        elif k > 0:
            x = ct[k:]
            y = rmssd[:-k]
        else:
            x = ct
            y = rmssd

        valid = ~np.isnan(x) & ~np.isnan(y)
        x = x[valid]
```

```
        y = y[valid]

        if len(x) > 3:
            sx = np.std(x)
            sy = np.std(y)
            if sx == 0 or sy == 0:
                corr = np.nan
            else:
                corr = np.corrcoef(x, y)[0, 1]
        else:
            corr = np.nan

        results.append((k, corr))

    return results



def main():
    parser = argparse.ArgumentParser()
    parser.add_argument("--data_dir", required=True, help="Cartella contenente file
RR in formato TXT")
    args = parser.parse_args()

    all_ct = []
    all_rmssd = []
```

```
all_rows = []

for file in Path(args.data_dir).glob("*.txt"):
    df = load_rr_file(file)
    windows = make_windows(df)
    metrics = compute_metrics(windows)

    for row in metrics:
        row["subject_id"] = file.stem
        all_rows.append(row)
        all_ct.append(row["c_t"])
        all_rmssd.append(row["rmssd_ms"])

ct = np.array(all_ct, dtype=float)
rm = np.array(all_rmssd, dtype=float)

valid = ~np.isnan(ct) & ~np.isnan(rm)
ct_valid = ct[valid]
rm_valid = rm[valid]

print("Numero finestre valide:", len(ct_valid))

if len(ct_valid) > 3:
    print("Correlazione globale:", np.corrcoef(ct_valid, rm_valid)[0, 1])
else:
```

```
        print("Dati insufficienti per correlazione globale")

    print("Lag:", lag_correlation(ct, rm))

    if all_rows:
        out_df = pd.DataFrame(all_rows)
        out_df.to_csv("life_omega_results.csv", index=False, encoding="utf-8")
        print("File salvato: life_omega_results.csv")
    else:
        print("Nessun risultato esportato")


if __name__ == "__main__":
    main()
```

APPENDICE B — FORMATO FILE RR (TXT)

Il file di input deve contenere una singola colonna di intervalli RR espressi in millisecondi.

Esempio:

820

830

810

805

790

780

770

760

750

740

APPENDICE C — FORMATO FILE JSON

Esempio di struttura logica di una finestra di analisi:


```
{
  "subject_id": "000",
  "window_start_s": 0.0,
  "window_end_s": 300.0,
  "rmssd_ms": null,
  "sdnn_ms": null,
  "energy_proxy": null,
  "spectral_entropy": null,
  "delta_s_dis": null,
  "c_t": null
}
```


Il formato JSON è utile per scambio dati, archiviazione o integrazione con altri strumenti, ma non è necessario per l'esecuzione dello script principale.

APPENDICE D — FORMATO FILE CSV DI OUTPUT

Lo script genera un file CSV denominato:

life_omega_results.csv

Il file contiene, per ogni finestra valida:

- identificativo soggetto
- inizio finestra
- fine finestra
- RMSSD
- SDNN
- energy proxy
- spectral entropy
- DeltaS_dis
- C_T

APPENDICE E — NOTA TECNICA SULLA CORRELAZIONE CON RITARDO

La correlazione globale tra C_T e RMSSD viene calcolata solo su finestre valide.

L'analisi del ritardo temporale, invece, viene calcolata sulle serie originali non compattate. I valori non validi vengono filtrati solo all'interno di ciascun lag, dopo aver mantenuto l'allineamento temporale tra finestre corrispondenti.

Questa distinzione è necessaria per evitare la perdita della struttura temporale del segnale.

APPENDICE F — NOTA LEGALE FINALE

Il codice allegato è parte integrante del presente trattato come strumento di verifica operativa del modello descritto nel testo.

Ogni uso scientifico, didattico o divulgativo deve citare l'autore del trattato e mantenere il riferimento all'opera completa.

L'eventuale utilizzo del codice su dataset pubblici resta soggetto alle condizioni d'uso e alle licenze dei relativi archivi di provenienza.

CHIUSURA

Il presente documento si chiude qui nella sua forma testuale e operativa.

Le verifiche ulteriori dipendono esclusivamente dall'esecuzione empirica dello script sui dataset considerati e dall'analisi dei risultati ottenuti.